Pattern Anal Applic (2016) 19:1–10
DOI 10.1007/s10044-015-0511-1

ORIGINAL ARTICLE

Tree-based binary image dissimilarity measure with meta-heuristic optimization

Bartłomiej Zieliński[1] · **Marcin Iwanowski**[1]

Received: 12 March 2014 / Accepted: 14 August 2015 / Published online: 30 August 2015

Abstract In this paper, we present a method of evaluating binary image dissimilarity based on tree representation and heuristic optimization. Starting from the image, a graph structure of a binary tree is constructed that splits the set of image foreground pixels into consecutive subsets attached to tree nodes. Next, instead of comparing two images themselves, one compares the trees and expresses image dissimilarity as tree dissimilarity, which can be characterized by a nonlinear function. The goal is to find its minimum, as it corresponds with the best match of compared trees. Searching for the minimum would be ineffective with analytical optimization methods. Hence, we have approached the issue with three meta-heuristic algorithms, namely genetic algorithm, particle swarm optimization (PSO) and simulated annealing. The presented results show that PSO achieved the best results. The proposed method is compared with other binary image comparison approaches. The performed tests that are described in the paper show that it outperforms its competitors and can be successfully applied to compare binary images.

Keywords Binary image dissimilarity measure · Binary image tree representation · Meta-heuristic optimization · Genetic algorithm · Particle swarm optimization · Simulated annealing

✉ Bartłomiej Zieliński
bartlomiej.zielinski@gmail.com

Marcin Iwanowski
marcin.iwanowski@ee.pw.edu.pl

[1] Institute of Control and Industrial Electronics, Warsaw University of Technology, ul. Koszykowa 75, 00-662 Warsaw, Poland

1 Introduction

The idea presented in this paper follows the approach of extracting and comparing image features to calculate image dissimilarity, which is understood as the extent of how two images are unlike, with scale, rotation and translation invariance. The image (dis)similarity is understood in visual terms. No semantic issues are raised. The proposed method is as follows. For a binary image, a tree (a connected, acyclic graph) is created. The tree represents the image. The dissimilarity between two images is expressed by the degree of dissimilarity of the trees, constructed for the images. To determine the dissimilarity value, the trees have to be matched, which is a nonlinear optimization problem. We address it with the use of meta-heuristic algorithms.

Determining the dissimilarity is based on tree representation of binary images and unlikeness between two images is expressed as unlikeness of the trees constructed upon the images [40–43]. In this paper, the proposed image comparison method has been juxtaposed with reference algorithms. Moreover genetic algorithm, particle swarm optimization and simulated annealing were analyzed as optimization methods for the need of tree matching in the presented approach.

Heuristic optimization methods (often biologically inspired) have constituted an important share in optimization studies recently. The interest in this area has been intensified by successful research, which has yielded many efficient algorithms. They are often used to cope with problems, which are too challenging for standard approaches. Namely, meta-heuristic methods effectively deal with nonlinear, multiple optima functions, with no derivative information, for both continuous and discrete domains. In this paper, we evaluate three meta-heuristic optimization

algorithms on a function, which calculates dissimilarity of binary images.

The paper is organized as follows. In the next section, a short survey of previous works in binary image similarity is given. In the next one, the proposed tree-based binary image dissimilarity measure, which underlies the experiments, is described. In Sect. 4, we briefly present the examined optimization algorithms. The experimental results are shown and discussed in Sect. 5. Conclusions are drawn in Sect. 6.

2 Previous works

The problem of determining binary image similarity is an important one in the domain of image processing. The aim is to design a solution robust to basic image transformations, e.g., geometrical ones: scaling, rotation and translation. Moreover, the approach should be robust to basic image transformations. Many different measures have been proposed so far. Some of them are based on the assumption that considered images are of the same size and orientation, and comparison of corresponding pixels is performed. Such methods are usually simple and computationally inexpensive; however, they are not invariant to image modifications. For this reason, more sophisticated methods have been developed. Most of them take advantage of extracting and comparing image features like edges, shapes, etc.

The issue of shape similarity is addressed in, e.g., [3] and [37]. In [13], the inner distance is introduced to measure the shape similarity. The inner distance is defined as the length of the shortest path within the shape boundary to classify shape images. The approach presented in [22] is aimed at handling partially visible shapes. Some methods make use of curve alignment techniques [31]. In [21], curve correspondence is applied to distinguish between shapes. In [1] and [27], the problem is approached with the so-called curvature scale space. The paper [19] introduces a representation based on a recursive division of a shape. Some methods operate on edge images [30] or contours of objects [23]. Others take advantage of curvature trees [2], contour point histogram [34], deformable potential of contour [38] or curve adjustment procedure based on their minimal deformation cost [39]. There is a number of algorithms which are based on Hausdorff distance [10, 16] to compare binary images. These algorithms generally calculate inter-pixel distances between images. The original solution is sensitive to noise and some research [4, 29] was aimed at reducing this shortcoming. An adaptive measure of local Hausdorff distances between images is proposed in [5]. In [15], the image dissimilarity is determined with the use of moment invariants. Both Hausdorff and moment invariant approaches are universal and possible to be applied for various kinds of binary images.

3 Tree-based image dissimilarity

A binary image is usually perceived as a rectangular matrix of two-valued scalars. The main idea of the proposed method is to represent a binary image as a tree. A tree T is understood to be a connected graph with no cycles. Trees considered in the paper are rooted ones, which means that there exists one node in the tree designated as a root. A parent of a node is the node connected to it on the path to the root. $|T|$ signifies the order (number of nodes) of T.

We present an image dissimilarity measure which in general can be applied to both binary [41] and gray-tone images [42]. To determine the dissimilarity of two images, trees created upon the images are compared. This is done in the process of adjusting one tree to the other. The problem itself is challenging and cannot be solved with the use of standard analytical optimization methods. Therefore, searching for the best match of the trees is performed by means of meta-heuristic optimization. This issue will be discussed in Sect. 4; the current one is focused on tree representation and comparison.

3.1 Constructing the tree

The tree construction algorithm works iteratively. Starting from the initial set of all image foreground pixels of a given binary image, its consecutive subsets are extracted. Their characteristic elements are than assigned to tree nodes. Let us define the set of image pixels $S = \{p : I(p) = u\}$, where p stands for pixel coordinates of an image I, and u is a value of foreground pixels. In successive steps, subsets of the input set of foreground pixels are created. Until the desired tree level is reached, subsets are recursively determined. For every set, a characteristic element is calculated. The tree is constructed by connecting the characteristic elements. As they are bound with sets of pixels, the elements remain in parent–child relationship as well and a tree arises by linking the elements which fulfill the relation. To apply the algorithm, one has to provide the input set of pixels S, a function f which determines the characteristic element, a criterion c which allows to construct subsets upon a set and the height of the output tree. Algorithm 1 presents the idea.

Algorithm 1 Constructing Tree T

```
procedure T ← Set2Tree(f, c, S, h)
    root(T) ← f(S)
    if h > 0 then
        (S1, S2) ← c(S)
        for all i ∈ ⟨1, n⟩ do
            Ti ← Set2Tree(f, c, Si, h − 1)
            parent root(Ti) ← root(T)
        end for
    end if
    return T
end procedure
```

To determine the characteristic element of a set, function f computes the arithmetic mean of all $p \in S$. In other words, f calculates coordinates of the centroid (center of gravity) of all pixels in S. Each tree node is thus characterized by an appropriate centroid. A criterion c allowing to create two subsets of S is defined as follows. Let $\rho(o_1, o_2)$ be the Euclidean distance between pixels' coordinates o_1 and o_2. Let $m = \text{median}(\{\rho(p, f(S)) : p \in S\})$. Presented in another way, m stands for a median distance between pixels in S and its centroid. Then, $c(S) = (S_1, S_2)$, where $S_1 = \{p \in S : \rho(p, f(S)) \leq m\}$ and $S_2 = \{p \in S : \rho(p, f(S)) \geq m\}$. In this way, S_1 contains pixels closer (or equally distant) to the centroid than m and S_2 contains pixels farther (or equally distant) from the centroid than m. Alternatively, instead of computing medians for consecutive sets of pixels, in the very beginning the quantiles may be calculated. Then, successive sets of pixels are determined with regard to the values of appropriate quantiles [40].

As there are two subsets created in a single step of our procedure, the output tree T is a perfect binary one. The algorithm successively applies function f to determine the characteristic elements for consecutive sets extracted with use of criterion c. The process stops when the tree is constructed to the desired level. Figure 1 illustrates the process of building a tree.

Apart from a centroid, with every set we associate information about pixel distribution of the set. The information comprises average pixel distance from the centroid and standard deviation of pixel distance from the centroid. This ensures comprehensive characterization of the set.

The trees created with the use of the above algorithm describe the content of images in such a way that for different images, depicting different content, corresponding centroids as well as pixel distribution values differ. On the other hand, centroids and pixel distribution values of corresponding nodes of trees created upon similar images remain similar, due to similar subsets returned by criterion c. Therefore, image similarity (or dissimilarity) can be expressed by tree similarity (or dissimilarity). We can estimate the extent of dissimilarity between two trees and therefore conclude upon the degree of difference between the images. Examples of trees created upon four various binary shapes are shown in Fig. 2.

3.2 Comparing the trees

Following the main idea of the proposed approach, to find the dissimilarity of two binary images one computes the dissimilarity of trees obtained for images being compared. Since the way of creating the tree results in a strict structure of the tree, while investigating differences between two trees, nodes can be compared pairwise. This is due to the fact that every node has its corresponding one in another tree. Difference between the trees is computed as a sum of differences between pairs of nodes. This way, the tree dissimilarity is expressed as a single scalar value.

Let T_1 and T_2 be trees of height h_1 and h_2, respectively. The difference between the trees can be computed as:

$$D = \sum_{i=1}^{l} \delta_i, \tag{1}$$

where δ_i stands for the difference for a pair of nodes i, $l = 2^{h+1} - 1$ with $h \in \mathbb{N}_+$ and $h \leq \min(h_1, h_2)$, and l is the number of nodes. The difference δ_i can be viewed as a "work" needed to translate a centroid of a node i to the position of a centroid of a corresponding node of another tree and vice versa. The term "work" is an analog to the physical quantity, which is directly proportional to force and the Euclidean distance between the centroids of a pair of corresponding nodes. The force is proportional to the number of pixels z_{ei} upon which the centroid for a node i in tree T_e is calculated. This reasoning leads to the formula:

$$\delta_i = \left(\frac{z_{1i}}{Z_1} + \frac{z_{2i}}{Z_2}\right)(d_i + \widehat{a}_i + \widehat{s}_i), \tag{2}$$

where $Z_e = \sum_{i=1}^{l} z_{ei}$, d stands for the distance, $\widehat{a}_i = |\widehat{a}_{1i} - \widehat{a}_{2i}|$ and $\widehat{s}_i = |\widehat{s}_{1i} - \widehat{s}_{2i}|$, with $\widehat{a}_{ei}$ meaning average pixel distance from the centroid i in tree T_e and $\widehat{s}_{ei}$ symbolizing standard deviation of pixel distance from the centroid i in tree T_e.

The above approach may be used directly only if the trees represent images of the same size and orientation. In case of images modified with the use of rotation or scaling, the trees must be matched prior to the computation of dissimilarity. To do this, first of all, trees are aligned, so that their roots are positioned in the origin of the coordinate system. It is achieved by translating all the nodes of a tree by the same vector applied to translate the root of the tree to the origin of the coordinate system. This ensures translation invariance of the trees. Then, tree T_1 is rotated and

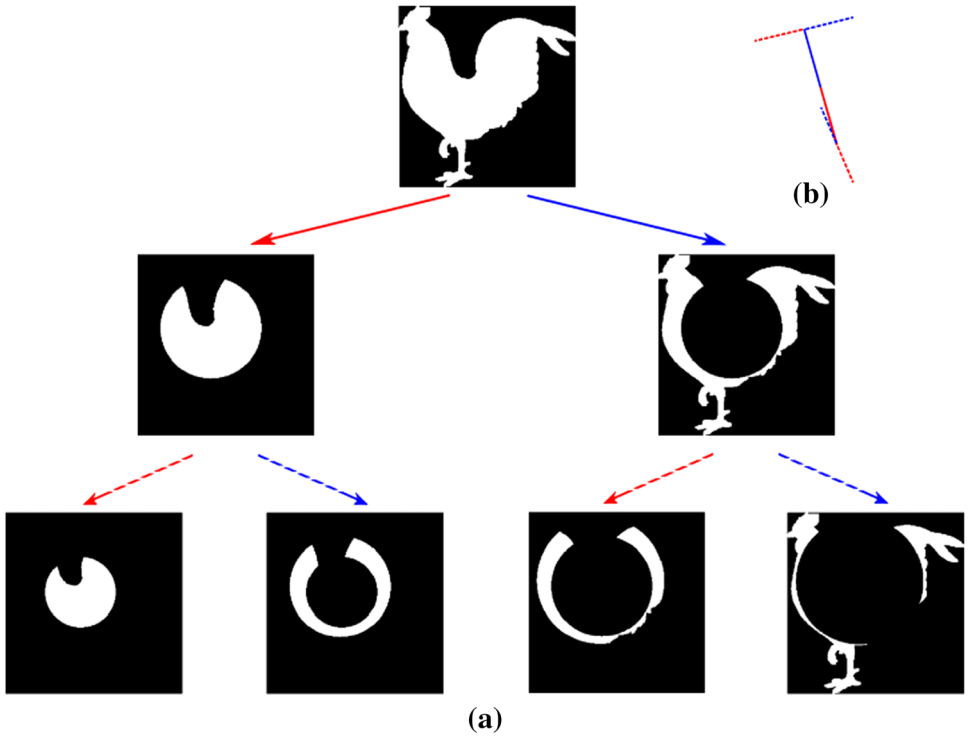

Fig. 1 The way of creating a tree. In **a**, consecutive sets returned by criterion c are presented. The images are ordered as a tree, in accordance with the way the procedure works. The topmost image is the input image. Left child images (*red arrows*) are composed of pixels which are closer or equally distant to the centroid of the parent image. Right child images (*blue arrows*) are composed of pixels which are farther or equally distant to the centroid. For every set, a characteristic element is calculated. The elements are bound with corresponding nodes of a tree. Every element is linked with a characteristic element of a parent node. This way, the final tree (**b**) arises. The presented tree of height $h = 2$ is enlarged in relation to the images in **a**. The *color* and the style of tree edges correspond to the *color* and the style of *arrows* in **a** (color figure online)

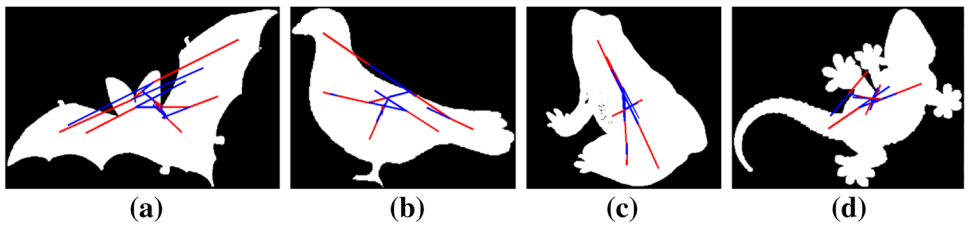

Fig. 2 Different binary images imply different trees. Test images of MPEG7 database [28]: **a** bat, **b** bird, **c** frog and **d** lizard exemplify the statement. The trees are of height $h = 4$. Left edges (connecting the sets with subsets of closer pixels) are drawn in *red*, while right edges (connecting the sets with subsets of farther pixels) are drawn in *blue* (color figure online)

scaled. This is achieved by transforming coordinates of all nodes of the tree to the polar coordinate system, in which each point is determined by an angle from a fixed direction and a distance from the pole. For every node of the tree, the angle is increased by α and the distance is multiplied by s. This way, a new tree T_1 arises. Next, the difference D between T_1 and T_2 is calculated.

Let us assume that the image upon which the tree T_2 has been constructed has been scaled, so that the distances in the image increased by factor κ and the number of pixels by factor κ^2. The tree constructed for that image would be a version scaled by κ of an original tree. Then the formula goes as follows:

$$\begin{aligned}\delta_i &= \left(\frac{z_{1i}}{Z_1} + \frac{\kappa^2 z_{2i}}{\kappa^2 Z_2}\right)\kappa(d_i + \widehat{a}_i + \widehat{s}_i) \\ &= \left(\frac{z_{1i}}{Z_1} + \frac{z_{2i}}{Z_2}\right)\kappa(d_i + \widehat{a}_i + \widehat{s}_i),\end{aligned} \tag{3}$$

which is dependent on κ. Obviously, a method with no dependence on scale is desired. Then the formula would have to be as follows:

$$\delta_i = \left(\frac{z_{1i}}{Z_1} + \frac{\kappa^2}{\kappa^2}\frac{z_{2i}}{Z_2}\frac{1}{\kappa}\right)\kappa(d_i + \widehat{a}_i + \widehat{s}_i). \tag{4}$$

Let us examine the $\frac{z_{2i}}{Z_2}$ component. It cannot be divided by κ, whose value is unknown. What is possible is to raise Z_2 to the power of 1.5 and then the scaled component would be

$$\frac{\kappa^2 z_{2i}}{(\kappa^2 Z_2)^{1.5}} = \frac{\kappa^2 z_{2i}}{\kappa^2 Z_2}\frac{1}{\kappa}. \tag{5}$$

The first component $\frac{z_{1i}}{Z_1}$ cannot be divided by κ either. However, the value s' by which the tree T_1 has been scaled in relation to the tree T_2 is known. The scale equals $s' = s\,\kappa$, where s is the scale difference between the compared images and is obviously dependent on κ. Therefore, we achieve:

$$\frac{z_{1i}}{Z_1}\frac{1}{s\,\kappa}, \tag{6}$$

which eliminates, in the first component, the relation between δ_i and κ. However, for an image scaled by factor x, upon which the tree T_1 has been created, the first component would equal

$$\frac{x^2 z_{1i}}{x^2 Z_1}\frac{x}{s\,\kappa}, \tag{7}$$

which in turn introduces the dependence on x. Removing the dependence is possible by raising Z_1 to the power of 1.5. Then for an image scaled by x, for which the tree T_1 has been built, the component would equal

$$\frac{x^2 z_{1i}}{(x^2 Z_1)^{1.5}}\frac{x}{s\,\kappa} = \frac{x^2 z_{1i}}{x^3 {Z_1}^{1.5}}\frac{x}{s\,\kappa} = \frac{z_{1i}}{{Z_1}^{1.5}}\frac{1}{s\,\kappa} \tag{8}$$

and the final formula takes the following form:

$$\delta_i = \left(\frac{z_{1i}}{Z_1^{1.5}}\frac{1}{s} + \frac{z_{2i}}{Z_2^{1.5}}\right)(d_i + \widehat{a}_i + \widehat{s}_i), \tag{9}$$

which ensures invariance to scaling.

The goal is to find an angle α and a scale s such that the the difference D is minimal. Then, D specifies the dissimilarity of trees, and thus the dissimilarity of images. When comparing trees created upon the same images even if rotated or scaled, D equals 0 (or almost 0 due to rotation and scaling approximation). The difference between the trees increases along with the growth of the difference between the images and is—theoretically—unlimited. Therefore, $D \in \langle 0, \infty)$.

A significant feature of the method is that the geometrical transformation of an image refers to the transformation of the tree constructed upon the image. In effect, the trees created with the use of the presented approach can be adjusted by any geometrical transformation, e.g., a mirror flip. In such case, there will obviously be a need to optimize some additional parameters, specific for a particular transformation.

4 Optimization algorithms

In the transformation model introduced in the previous section, two parameters are considered, namely angle α and scale s. Matching the trees with regard to these parameters is an optimization problem. The goal is to optimize the function expressed by the summation (1). The search space is a two parameters plane. In general, however, adjustment of several parameters may be required. Moreover, the function is nonlinear, it can have multiple minima and calculating its derivative is a very expensive task in computational terms. For such challenging functions, heuristic optimization algorithms, rather than the classical ones, are usually the best choice. Accordingly, in this research, the heuristic optimization approaches were applied. There is a constant necessity for deeper comprehension and further improvement of this type of method. Both external, computational needs and bottom-up algorithm examination appoint directions of advancement in the field of optimization. Heuristic optimization is applied in the current study to find the best match of two trees created upon binary images. This matching is necessary to compute the dissimilarity between the trees in view of evaluating the dissimilarity of two underlying images. In this research, we used three meta-heuristic optimization algorithms, namely, genetic algorithm, particle swarm optimization and simulated annealing. In the remainder of this section, the general view of applied optimization methods has been presented. Detailed descriptions, algorithm variants as well as parameter selection hints are available in a vast literature on the subject (e.g. [6, 7, 11, 12, 14, 18, 20, 26, 33]).

4.1 Genetic algorithm

Genetic algorithm (GA) [12, 14, 26] is a classic meta-heuristic optimization approach. It follows the concept of population advancement based on natural evolution. A population composed of individuals (also called chromosomes) evolves to find the best argument value of an objective function. Individuals undergo mechanisms derived from evolution, to lead the population to better results. The algorithm works as follows. At first, the population is randomly initiated in the search space. Then, in each iteration, chromosomes, each of which encodes a solution, are subjected to selection, gene recombination and mutation. These imitate evolution of a real, biological species. In effect, the next generation replaces the previous one. Selection determines pairs of parent chromosomes,

upon which children arise by means of gene crossover. Generally, the better is the value of the fitness function for a particular individual, the greater are the chances for it to become a parent. Crossover operator defines how genes of parent individuals are recombined. Mutation decreases the risk of the population getting stuck in a local minimum. The main parameters are population size P, maximal number of generations G and fraction of the population which undergoes crossover and mutation.

In the issue of tree dissimilarity minimization, each chromosome encoded a scale s and an angle α. The population evolved in search for the optimal values of these parameters.

4.2 Particle swarm optimization

Particle swarm optimization (PSO) [18, 33] is a well-known optimization algorithm. It is an evolutionary computation method, which imitates social behavior of a swarm of creatures. The idea benefits from advancement of a population of particles, which rove in the search space. The position of each particle represents a solution. Particles adjust their movement parameters according to their own experience as well as the experience of other particles of the swarm. A basic variant of PSO works as follows. First of all, the population of particles is initiated. Particles, denoted below as x, are randomly positioned over a search space and are assigned random velocities. At iteration $k+1$, the position of a particle i is updated according to the formula:

$$x_{k+1}^i = x_k^i + v_{k+1}^i \tag{10}$$

with the velocity v_{k+1}^i calculated as follows:

$$v_{k+1}^i = w_k\, v_k^i + c_{1k}\, r_1 \left(p_k^i - x_k^i\right) + c_{2k}\, r_2 \left(p_k^g - x_k^i\right), \tag{11}$$

where w_k stands for inertia weight and c_{1k} and c_{2k} stand for cognitive attraction and social attraction parameters, respectively. p_k^i is the best solution particle i ever found, p_k^g is the best solution ever found by any particle of the swarm, and r_1 and r_2 are random numbers uniformly distributed on the interval (0, 1). Inertia weight w_k as well as attraction parameters c_{1k} and c_{2k} can be constant or may be changed in the course of the run. The algorithm works and improves the solution until the stopping criterion is met, e.g., satisfactory solution is achieved, inconsiderable solution improvement is negligible or a maximum number of iterations is reached. PSO has gained an exceptional interest. There are many works introducing modifications to increase the performance of PSO and providing analysis of the algorithm, e.g., [7, 8, 11, 25]. Most of them relate to updating particle velocity.

In search of the best tree adjustment, we assumed that each particle's position represents a scale s and an angle α. Velocity expresses the dynamics of parameter adjustment. Swarm proceeded to find the optimal configuration of the two parameters.

4.3 Simulated annealing

Simulated annealing (SA) [6, 20] is another meta-heuristic, global optimization approach. It is inspired by the process of annealing in metallurgy, where heating and cooling down the alloy is performed, to achieve the best structure of the material, therefore minimizing its energy. Heating shoves the molecules out from their initial positions. This prevents the algorithm getting stuck in a local minimum. On the other hand, cooling lets them go into the state of lower energy then before, which means they head toward an optimal solution. In each step of the algorithm, a new point is randomly generated. Its distance to the current point is based on a probability distribution, whose scale is proportional to the temperature. The current solution is replaced by a new one, when the latter lowers the objective (cooling down the material). However, with a certain probability, a new point may be accepted, even if it raises the objective (heating). The temperature is decreased in the run of the algorithm. This way, the extent of search is reduced in the final stages. The main parameters are the initial temperature and the cooling function, which determine the way of decreasing the temperature.

Similarly as in the other approaches, we optimized our objective function with respect to scale s and angle α.

5 Experiments

As it has been mentioned, the tree-based method allows to determine the dissimilarity between any pair of binary images. The following section presents a comparison of the proposed approach with two reference algorithms, namely the Hu moment invariant method [15] and a modified Hausdorff distance method [10]. All three compared methods are general purpose ones, able to deal with different types of binary images, and their outcome can be presented as a dissimilarity value.

The Hu moment invariants approach allows to calculate image characteristic (as a set of 7 numbers) invariant to scaling, translation and rotation. The methods based on Hausdorff distance are sensitive to such transformations. It is possible to deal with the issue by translating, resizing, as well as determining orientation [9] and rotating of the compared images. However, it significantly increases the computational cost.

Fig. 3 Binary image variants

Moreover, another set of experiments was performed. For the tree-based approach, three meta-heuristic optimization algorithms were evaluated and juxtaposed.

5.1 Methodology and results

We created a dataset of binary images[1], based on the MPEG7 shape database [28]. Our dataset is composed of 50 classes of images. Every class contains an original object as well as its variants, obtained by binary morphological operators of opening and closing [32]. Therefore, there are 150 images in the dataset. Every morphologically modified image was scaled by a random factor in the range $\langle 0.5, 2\rangle$ and rotated by a random factor in the range $\langle 0, 2\pi\rangle$. Class example is presented in the Fig. 3. Upon the images, trees of height $h = 3$ were created. Evaluation procedure were as follows. For a given optimization algorithm, for its certain configuration, every possible pair of trees was compared. As a result, we obtained square matrix of dissimilarity values. If an effective optimization method was used, then the values calculated for objects of the same class were small, whereas those calculated for objects from among different classes were greater. To reason about the efficiency of the considered algorithm, we subjected the matrix of dissimilarity values to cluster analysis.

Clustering was performed with the use of k-medoids algorithm [17]. The method—on the basis of the dissimilarity matrix—partitions the set of observations into a predefined number of clusters k. It is similar to k-means algorithm [24, 36]. However, in contrast to k-means, k-medoids selects cluster centers from among the observations rather than calculates centroids. The chosen centers (called medoids) define clusters in such a way that each observation is assigned to the least dissimilar medoid. The objective of the algorithm is to find such a configuration of medoids for which the sum of dissimilarities between the observations and their centers is minimal. The parameter k was constant and equaled to 50—the number of classes of images.

[1] Available at http://www.isep.pw.edu.pl/~iwanowsm/binary.zip.

We express the reliability as the percentage of correctly clustered images. It may be difficult, however, to determine the percentage. Clustering algorithms deliver data grouped into clusters, but without labels. To determine the reliability, it is necessary to find the best cluster labeling. However with different observations within a single cluster, it is tricky to assign the labels. For every case, we aimed at finding such a label arrangement which led to the highest percentage.

The comparative study of the algorithms follows the reasoning that the better an optimization method performed, the higher was the outcome percentage result. The successive tables present results obtained for the considered algorithms, for different parameter configuration. The parameters were arranged in such a way that a maximal number of function evaluation was fixed. In effect, the results of comparable computational effort for every method were achieved.

For comparative purposes, the proposed approach was confronted with two well-known binary image comparison algorithms, namely the Hu moment invariants [15] and the modified Hausdorff approach [10]. The results achieved for the Hu method reached 38.40 % and for the modified Hausdorff approach 36.80 %.

5.2 Comparison of meta-heuristics for the tree-based approach

GA has been applied with different values of population size P, maximal number of generations G and number of chromosomes which underwent crossover. Selection was performed by stochastic universal sampling. Gene recombination was accomplished by uniform crossover. Mutation was executed in an adaptive way [35]. Table 1 presents the results.

We have applied the PSO algorithm with different values of population size P, maximal number of generations K, cognitive attraction c_{1k} and social attraction c_{2k}. Initial inertia weight $w_1 = 0.9$ and $w_k = w_1 - 0.4\,(k-1)/(K-1)$, which means that w_k decreased linearly from 0.9 at the first iteration to 0.5 at iteration K. PSO was made to

Table 1 Percentage results of GA for different parameters—population size, generations and number of chromosomes A' which undergo crossover

Generations	Population								
	5			10			15		
	A'			A'			A'		
	2	3	4	4	6	8	6	9	12
10	58.00	43.33	44.67	76.67	71.33	46.00	86.67	82.67	62.67
15	61.33	42.00	41.33	83.33	75.33	47.33	**88.00**	84.00	65.33
20	63.33	46.67	46.00	86.00	78.67	46.67	86.00	87.33	68.00

The largest value is emphasized in bold

Table 2 Percentage results of PSO for different parameters—population size, generations, cognitive attraction c_{1k} and social attraction $c_{2k} = 0.75$

Generations	Population								
	5			10			15		
	c_{1k}			c_{1k}			c_{1k}		
	0.25	0.50	0.75	0.25	0.50	0.75	0.25	0.50	0.75
10	90.67	86.67	94.67	94.00	91.33	94.00	96.67	96.67	97.33
15	86.00	87.33	92.00	96.00	98.00	94.00	94.67	**100.00**	98.67
20	92.00	94.67	90.00	99.33	94.67	96.67	96.67	97.33	**100.00**

The largest values are emphasized in bold

Table 3 Percentage results of PSO for different parameters—population size, generations, cognitive attraction c_{1k} and social attraction $c_{2k} = 1.25$

Generations	Population								
	5			10			15		
	c_{1k}			c_{1k}			c_{1k}		
	0.25	0.50	0.75	0.25	0.50	0.75	0.25	0.50	0.75
10	88.00	90.00	88.67	95.33	96.67	96.67	94.67	96.00	96.67
15	91.33	94.67	95.33	99.33	94.67	97.33	**100.00**	96.67	**100.00**
20	95.33	94.67	93.33	96.00	99.33	**100.00**	97.33	99.33	**100.00**

The largest values are emphasized in bold

Table 4 Percentage results of PSO for different parameters—population size, generations, cognitive attraction c_{1k} and social attraction $c_{2k} = 1.75$

Generations	Population								
	5			10			15		
	c_{1k}			c_{1k}			c_{1k}		
	0.25	0.50	0.75	0.25	0.50	0.75	0.25	0.50	0.75
10	87.33	86.00	90.67	92.00	98.00	95.33	97.33	98.67	96.67
15	89.33	92.00	90.00	94.67	97.33	98.67	94.67	**100.00**	96.67
20	92.67	94.00	90.00	**100.00**	**100.00**	99.33	**100.00**	96.67	**100.00**

The largest values are emphasized in bold

stop if there was no improvement in the objective function for 5 consecutive generations. Tables 2, 3 and 4 present the results.

The maximal number of function evaluations in a single run was fixed. This way, the algorithm was able to calculate the value of the function as many times as the tested population-based methods. The initial temperature was set to either $t_1 = 50$ or $t_1 = 100$ and its value at iteration k was calculated as $t_k = 0.95^k \times t_1$. Table 5 shows the results for both t_1 values and different numbers of objective function evaluations.

Experiments have been performed for small population sizes and numbers of generations, for both population-based methods. This limited the number of possible

Table 5 Percentage results of the SA approach for different initial temperature values and numbers of objective function evaluations N

Initial temperature	N						
	50	75	100	150	200	225	300
50	82.67	83.33	90.67	91.33	92.00	89.33	**97.33**
100	78.00	88.00	86.67	86.67	94.00	94.67	**97.33**

The largest values are emphasized in bold

evaluations of the objective function. For the simulated annealing approach, the number of the objective function evaluations was also limited. In effect, every approach was able to calculate the objective function at most the same number of times. This way, results are comparable. The experiments show that PSO achieved the best results, which were satisfactory for small population sizes and numbers of generations. Simulated annealing was slightly worse. Both methods significantly outperformed the genetic algorithm.

It is clearly visible that the genetic algorithm achieved better results for greater population size and number of generations. Crossover fraction also played a significant role in the performance of the method. As for values $f_1 = 0.7$ and $f_1 = 0.8$, the results have been similar, and increasing it to $f_1 = 0.9$ led to the degradation of the results.

In the PSO evaluation, the size of the population has had a positive impact on the outcome. On the other hand, for the number of generations, social and cognitive attraction, we have observed no obvious influence of the parameters on the final outcome.

As one might expect, for the simulated annealing approach, generally, greater number of possible objective function evaluations yielded better optimization results.

6 Conclusions

In this paper, a generic binary image comparison approach with the use of heuristic optimization has been presented. The method is based on image tree representation. In the process of matching the trees constructed upon two images, a dissimilarity value is obtained. The value expresses how the images are unlike. The proposed approach is invariant to translation, scaling and rotation, as well as robust to basic image transformations. In this research, we also found out how heuristic optimization algorithms perform with the tree adjustment issue for two transformation parameters—scaling and rotation. We have evaluated the genetic algorithm, the particle swarm optimization and the simulated annealing approach.The tests have been applied for a different parameter values of the considered methods. The results show the advantage of the particle swarm optimization and the simulated annealing algorithms over the genetic algorithm. Nevertheless, the best results have been obtained for the PSO approach. It is worth noticing that geometrical image transformation corresponds to the transformation of the tree. Therefore, the presented method can be naturally developed to be robust to any geometrical image transformation. Despite higher complexity which will occur in such a case (more variables to optimize), thanks to the proposed heuristic optimization approach, tree adjustment process may be performed using exactly the same scheme. The proposed methods for measuring binary image dissimilarity has also been compared in this paper with other dissimilarity measures, showing its superiority over them. All the above experiments have proven that the proposed method can be successfully applied to binary image comparison.

References

1. Abbasi S, Mokhtarian F, Kittler J (2000) Enhancing CSS-based shape retrieval for objects with shallow concavities. Image Vis Comput 18(3):199–211
2. Alajlan N, Kamel MS, Freeman G (2006) Multi-object image retrieval based on shape and topology. Signal Process Image Commun 21(10):904–918
3. Ankerst M, Kriegel H-P, Seidl T (1998) A multistep approach for shape similarity search in image databases. IEEE Trans Knowl Data Eng 10(6):996–1004
4. Baddeley AJ (1992) An error metric for binary images. Robust computer vision: quality of vision algorithms, proceedings of international workshop on robust computer vision, pp 59–78
5. Baudrier E, Nicolier F, Millon G, Ruan S (2008) Binary-image comparison with local-dissimilarity quantification. Pattern Recognit 41(5):1461–1478
6. Cerny V (1985) Thermodynamical approach to the traveling salesman problem: an efficient simulation algorithm. J Optim Theory Appl 45(1):41–51
7. Clerc M, Kennedy J (2002) The particle swarm—explosion, stability, and convergence in a multidimensional complex space. IEEE Trans Evol Comput 6(1):58–73
8. Clerc M (2010) Beyond standard particle swarm optimisation. Int J Swarm Intell Res 1(4):46–61
9. Crespo J, Aguiar P (2011) Revisiting complex moments for 2-D shape representation and image normalization. IEEE Trans Image Process 20(10):2896–2911

10. Dubuisson MP, Jain AK (1994) A modified Hausdorff distance for object matching. International conference on pattern recognition, pp 566–568
11. Eberhart R, Shi Y (2000) Comparing inertia weights and constriction factors in particle swarm optimization. In: Proceedings of the 2000 congress on evolutionary computation, pp 84–88
12. Goldberg D (1989) Genetic algorithms in search. Optimization and machine learning. Addison-Wesley Longman Publishing Co., Inc., Boston
13. Haibin L, Jacobs DW (2007) Shape classification using the inner-distance. IEEE Trans Pattern Anal Mach Intell 29(2):286–299
14. Holland J (1975) Adaptation in natural and artificial systems. University of Michigan Press, Michigan
15. Hu MK (1962) Visual pattern recognition by moment invariants. IRE Trans Inf Theory 8(2):179–187
16. Huttenlocher DP, Klanderman GA, Rucklidge WJ (1993) Comparing images using the Hausdorff distance. IEEE Trans Pattern Anal Mach Intell 15(9):850–863
17. Kaufman L, Rousseeuw P (1990) Finding groups in data: an introduction to cluster analysis. Wiley-Interscience, New York
18. Kennedy J, Eberhart R (1995) Particle swarm optimization. Proc IEEE Int Conf Neural Netw 4:1942–1948
19. Kim H-K, Kim J-D (2000) Region-based shape descriptor invariant to rotation, scale and translation. Signal Process Image Commun 16(1):87–93
20. Kirkpatrick S, Gelatt C, Vecchi M (1983) Optimization by simulated annealing. Science 220(4598):671–680
21. Latecki LJ, Lakmper R (2000) Shape similarity measure based on correspondence of visual parts. IEEE Trans Pattern Anal Mach Intell 22(10):1185–1190
22. Latecki LJ, Lakmper R, Wolter D (2005) Optimal partial shape similarity. Image Vis Comput 23(2):227–236
23. Linhui J, Kitchen L (2000) Object-based image similarity computation using inductive learning of contour-segment relations. IEEE Trans Image Process 9(1):80–87
24. MacQueen J (1967) Some methods for classification and analysis of multivariate observations. Fifth Berkeley symposium on mathematical statistics and probability, pp 281–297
25. Mendes R, Kennedy J, Neves J (2004) The fully informed particle swarm: simpler, maybe better. IEEE Trans Evol Comput 8(3):204–210
26. Michalewicz Z (1996) Genetic algorithms + data structures = evolution programs. Springer, London
27. Mokhtarian F, Abbasi S, Kittler J (1997) Efficient and robust retrieval by shape content through curvature scale space. World Sci Publ Ser Softw Eng Knowl Eng 8:51–58
28. MPEG7 CE Shape-1 Part B. http://imageprocessingplace.com/downloads_V3 /root_downloads/image_databases/MPEG7_CE-Shape-1_Part_B.zip. Accessed 19 Nov 2014
29. Paumard J (1997) Robust comparison of binary images. Pattern Recognit Lett 18(10):1057–1063
30. Prieto MS, Allen AR (2003) A similarity metric for edge images. IEEE Trans Pattern Anal Mach Intell 25(10):1265–1273
31. Sebastian TB, Klein PN, Kimia BB (2003) On aligning curves. IEEE Trans Pattern Anal Machine Intell 25(1):116–125
32. Serra J (1982) Image analysis and mathematical morphology. Academic Press, London
33. Shi Y, Eberhart R (1999) Empirical study of particle swarm optimization. In: Proceedings of the 1999 congress on evolutionary computation, pp 1945–1950
34. Shu X, Wu X-J (2011) A novel contour descriptor for 2D shape matching and its application to image retrieval. Image Vis Comput 29(4):286–294
35. Srinivas M, Patnaik LM (1994) Adaptive probabilities of crossover and mutation in genetic algorithms. IEEE Trans Syst Man Cybern 24(4):656–667
36. Steinhaus H (1956) Sur la Division des Corps Materiéls en Parties. Bull Acad Polon Sci C1.III
37. Veltkamp RC (2001) Shape matching: similarity measures and algorithms. SMI 2001 international conference on shape modeling and applications, pp 188–197
38. Xu C, Liu J, Tang X (2009) 2D shape matching by contour flexibility. IEEE Trans Pattern Anal Mach Intell 31(1):180–186
39. Younes L (1999) Optimal matching between shapes via elastic deformations. Image Vis Comput 17(5):381–389
40. Zieliński B (2013) Division trees and their application to digital image comparison. PhD dissertation (in Polish), Warsaw University of Technology
41. Zieliński B, Iwanowski M (2013) Binary image comparison with use of tree-based approach. Adv Intell Syst Comput 184:171–177
42. Zieliński B, Iwanowski M (2012) Comparing image objects using tree-based approach. Lecture Notes Comput Sci 7594:702–709
43. Zieliński B, Iwanowski M (2013) Heuristic optimization algorithms for a tree-based image dissimilarity measure. Adv Intell Syst Comput 226:91–100